

# APPENDIX A

1c542 U.S. PRO

09/365517



```

/*****\
 *
 * Copyright (c) Schlumberger Technologies 1999. All rights reserved.
 * No part of this program may be photocopied, reproduced, or
 * translated to another programming language without the prior
 * written consent of Yield Enhancement Systems.
 *
/*****\

/*****
Returns:
0 for success
-1 for functional failure
-2 if algorithm fails to find acceptable threshold
*****/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <mp.h>
#include <image.h>
#include <SLBapi.h>
#include <pgm.h>
#include <morph1.h>
#include <scatter2d8.h>

#define Sqrt2 1.414213562373 /*sqrt(2)*/
#define MIN_DIST 1 /*min image difference used as the
beginning of 1D profiles*/
#define MVA_SIZE 21 /*size of moving average window*/
#define MVA_RATIO 0.9 /*allowable shrinkage in 1d profile
wrt its mva*/
#define MIN_DATA_SIZE 80 /*min # of data points to use curve
fitting*/
#define MX_SIZE (3 << GRAY_SCALE_BITS) /*size of the data matrix for curve
fitting*/
#define HIST_MORPH_PASSES 1 /*no. of passes for open/close
operations*/
#define SIGMA_MAX 5 /*enable removal of outliers if sigma
of fit is bigger than this*/
/*#define MIN_THR 10 /*min thr below which no difference
is called a defect*/
#define EPSILON 0.1 /*min value for det (AtA)*/
#define ALG_FITMASK 1 /*detection alg using cleaned 2D hist
as a mask*/

static void hist2D8(unsigned char *a, unsigned long tcols_a,
unsigned char *b, unsigned long tcols_b,
unsigned long ncols, unsigned long nrows,
unsigned char *table);

static OneD_dist_profile distmax, distmin;

/*****\

int Find2DHistoOutliers(OIP_byte_image *pSrcImage1,
OIP_byte_image *pSrcImage2,
OIP_byte_image *pDstImageResult,
PIXEL_diff_para PixelDiffPara,
unsigned char *hist,
unsigned char *hist2DBitMap,
unsigned long FovID,
long shorttest)
{
int rc;
unsigned char *aImage1, *aImage2, *aResult; /* ROI pointer */
const long Passes = HIST_MORPH_PASSES; /*1=3x3 open operation, 2=5x5, etc*/
char name[128];

```

```

/* check if the ROIs have the same size */
if ((pSrcImage1->Info.RegWidth != pSrcImage2->Info.RegWidth)
    || (pSrcImage1->Info.RegWidth != pDstImageResult->Info.RegWidth)
    || (pSrcImage1->Info.RegHeight != pSrcImage2->Info.RegHeight)
    || (pSrcImage1->Info.RegHeight != pDstImageResult->Info.RegHeight))
    return -1;

/* determine ROI pointers */
aImage1 = pSrcImage1->BufPtr
        + pSrcImage1->Info.RegX0
        + pSrcImage1->Info.RegY0 * pSrcImage1->Info.Width;

aImage2 = pSrcImage2->BufPtr
        + pSrcImage2->Info.RegX0
        + pSrcImage2->Info.RegY0 * pSrcImage2->Info.Width;

aResult = pDstImageResult->BufPtr
        + pDstImageResult->Info.RegX0
        + pDstImageResult->Info.RegY0 * pDstImageResult->Info.Width;

memset(hist, 0, GRAY_LEVELS * GRAY_LEVELS * sizeof(unsigned char));
hist2D8(aImage1, pSrcImage1->Info.Width,
        aImage2, pSrcImage2->Info.Width,
        pDstImageResult->Info.RegWidth,
        pDstImageResult->Info.RegHeight,
        hist);

if (shorttest)
{
    sprintf(name, "../data/2Dhisto_%ld.pgm", FovID);
    printf("Writing 2Dhisto to %s\n", name); fflush(stdout);
    rc = write_image(name, hist, 256, 256);
    if (rc) ERROR_RETURN(rc, "write 2Dhisto failed");
}

return 0;
}

/*****/
void hist2D8(unsigned char *a, unsigned long tcols_a,
            unsigned char *b, unsigned long tcols_b,
            unsigned long ncols, unsigned long nrows,
            unsigned char *table)

/*****/
a,b -- TCL of ROI (images)
tcols_a -- image width
ncols, nrows -- ROI size
table -- scatter table (a 256x256 image)
*****/
{
    unsigned long i, j;
    unsigned long skip_a, skip_b;

    unsigned long index;

    skip_a = tcols_a - ncols;
    skip_b = tcols_b - ncols;

    for (i = 0; i < nrows; ++i)
    {
        for (j = 0; j < ncols; ++j)
        {
            index = ((*a++) << GRAY_SCALE_BITS) + (*b++);
            table[index] = 1;
        }

        a += skip_a;
        b += skip_b;
    }
}

```